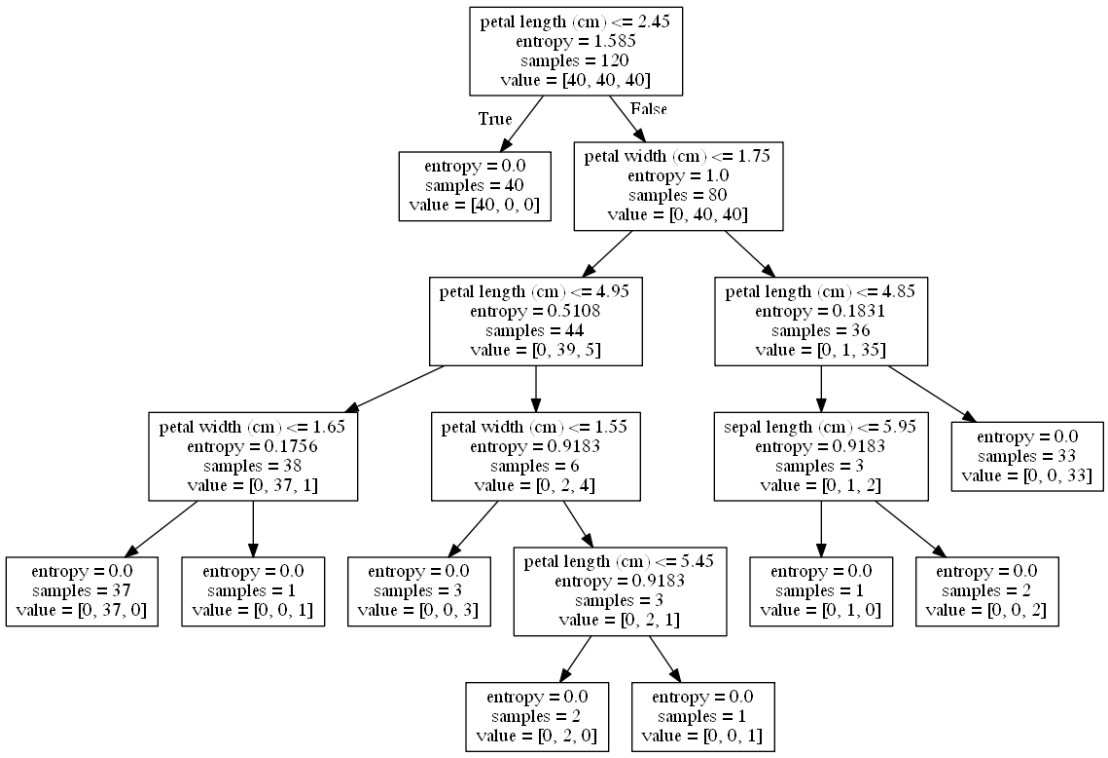
决策树案例

```
import numpy as np

from sklearn.datasets import load_iris

from sklearn.cross_validation import StratifiedShuffleSplit

from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix

iris = load_iris()
X = iris['data']
y = iris['target']

class_labels = iris['target_names'].tolist()
class_labels
['setosa', 'versicolor', 'virginica']

ds = np.column_stack([X,y])

# 分层划分,确保训练集和测试集的类别分布一致
strat_split = StratifiedShuffleSplit(ds[:,-1],test_size=0.2,
                                     n_iter=1,random_state=77)

for train_idx,test_idx in strat_split:
    X_train = ds[train_idx,:-1]
    y_train = ds[train_idx,-1]
    X_test = ds[test_idx,:-1]
    y_test = ds[test_idx,-1]

# 基于 gini
model = DecisionTreeClassifier()

model.fit(X_train,y_train)

# 在训练集上预测
y_predicted = model.predict(X_train)

accuracy_score(y_train,y_predicted)
1.0
```

```
confusion_matrix(y_train,y_predicted)
array([[40,  0,  0],
       [ 0, 40,  0],
       [ 0,  0, 40]])

print classification_report(y_train,y_predicted,
                            target_names=class_labels)
             precision    recall  f1-score   support

     setosa       1.00      1.00      1.00       40
 versicolor       1.00      1.00      1.00       40
  virginica       1.00      1.00      1.00       40

avg / total       1.00      1.00      1.00      120

# 在测试集上预测
y_predicted = model.predict(X_test)

accuracy_score(y_test,y_predicted)
0.9666666666666667

confusion_matrix(y_test,y_predicted)
array([[10,  0,  0],
       [ 0,  9,  1],
       [ 0,  0, 10]])

print classification_report(y_test,y_predicted,
                            target_names=class_labels)
             precision    recall  f1-score   support

     setosa       1.00      1.00      1.00       10
 versicolor       1.00      0.90      0.95       10
  virginica       0.91      1.00      0.95       10

avg / total       0.97      0.97      0.97       30

# 基于熵
model = DecisionTreeClassifier(criterion='entropy')

model.fit(X_train,y_train)

# 对测试集预测
y_predicted = model.predict(X_test)
```

```
accuracy_score(y_test,y_predicted)
1.0

confusion_matrix(y_test,y_predicted)
array([[10,  0,  0],
       [ 0, 10,  0],
       [ 0,  0, 10]])

print classification_report(y_test,y_predicted,
                            target_names=class_labels)
             precision    recall  f1-score   support

     setosa       1.00      1.00      1.00       10
 versicolor       1.00      1.00      1.00       10
  virginica       1.00      1.00      1.00       10

avg / total       1.00      1.00      1.00       30

# 各个特征的重要性
iris['feature_names']
['sepal length (cm)',
 'sepal width (cm)',
 'petal length (cm)',
 'petal width (cm)']

fi = model.feature_importances_

for i,fn in enumerate(iris['feature_names']):
    print "%s = %0.3f" % (fn,fi[i])

sepal length (cm) = 0.014
sepal width (cm) = 0.000
petal length (cm) = 0.668
petal width (cm) = 0.317

# 决策树
from sklearn.tree import export_graphviz

export_graphviz(model,out_file='dt.dot',
                feature_names=iris['feature_names'])
```


petal length (cm) <= 2.45
entropy = 1.585
samples = 120
value = [40, 40, 40]
True
False
entropy = 0.0
samples = 40
value = [40, 0, 0]
petal width (cm) <= 1.75
entropy = 1.0
samples = 80
value = [0, 40, 40]
petal length (cm) <= 4.95
entropy = 0.5108
samples = 44
value = [0, 39, 5]
petal length (cm) <= 4.85
entropy = 0.1831
samples = 36
value = [0, 1, 35]
petal width (cm) <= 1.65
entropy = 0.1756
samples = 38
value = [0, 37, 1]
petal width (cm) <= 1.55
entropy = 0.9183
samples = 6
value = [0, 2, 4]
sepal length (cm) <= 5.95
entropy = 0.9183
samples = 3
value = [0, 1, 2]
entropy = 0.0
samples = 33
value = [0, 0, 33]
entropy = 0.0
samples = 37
value = [0, 37, 0]
entropy = 0.0
samples = 1
value = [0, 0, 1]
entropy = 0.0
samples = 3
value = [0, 0, 3]
petal length (cm) <= 5.45
entropy = 0.9183
samples = 3
value = [0, 2, 1]
entropy = 0.0
samples = 1
value = [0, 1, 0]
entropy = 0.0
samples = 2
value = [0, 0, 2]
entropy = 0.0
samples = 2
value = [0, 2, 0]
entropy = 0.0
samples = 1
value = [0, 0, 1]